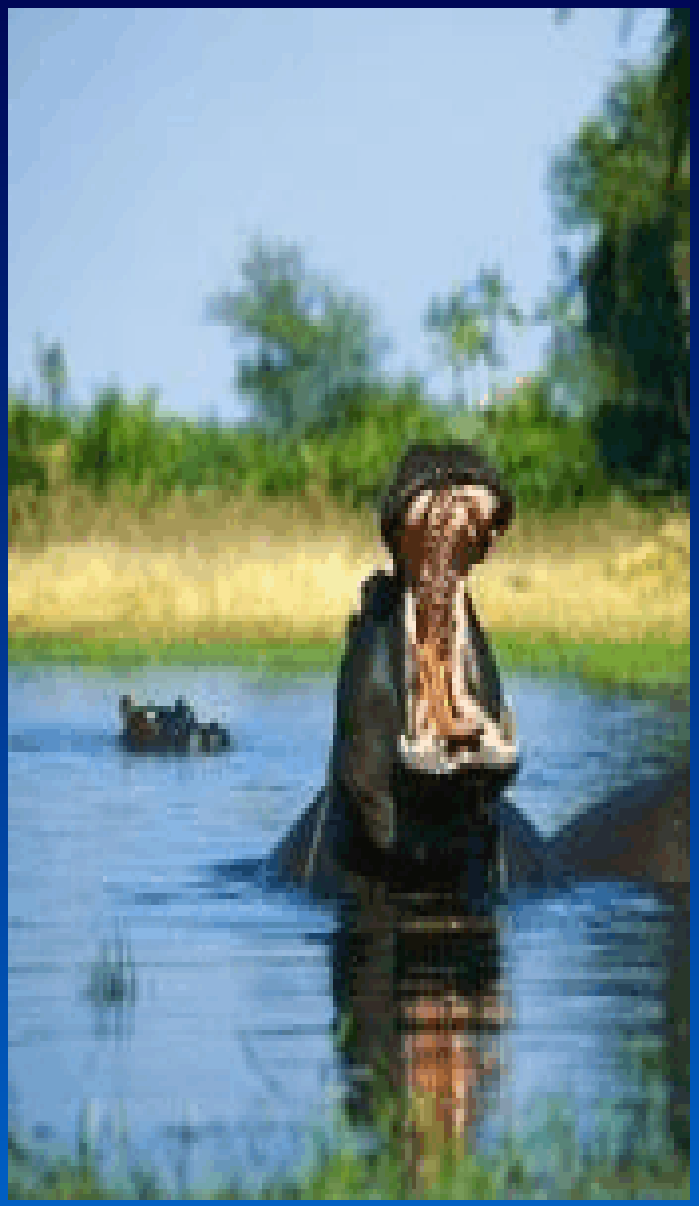


Practical Examples of Database Administration

Juliann Meyer
Sr. Hydrologist - Data Systems
Missouri Basin RFC

Hydrologic DBA Workshop
March 18-21, 2003

Querying the Sysmaster database

- **dbwho.sql**, show the database, who has it open, the workstation they are connected from, and the session id
- **dbsfree.sql**, list % free space in dbspaces
- **dblist.sql**, list all databases, owner & logging status
- **tabextent.sql**, list tables, # of extents & size of table
- **tabprof.sql**, table I/O activity

dbwho query

```
-- dbwho.sql
select  sysdatabases.name database,     --Database Name
syssessions.username,                   -- User Name
syssessions.hostname,                   -- Workstation
syslocks.owner sid                      -- Informix Session ID
from    syslocks, sysdatabases , outer syssessions
where   syslocks.tabname = "sysdatabases"
and  syslocks.rowidlk = sysdatabases.rowid
and  syslocks.owner = syssessions.sid
order by 1;
```

Example results of dbwho.sql

database	username	hostname	sid
fxatext	fxa	ds1-krf	171
fxatext	fxa	ds1-krf	172
gisrs	mgt	appl-krf	60548
gisrs	mgt	appl-krf	60549
hd_ob1krf	rds	ws2-krf	58969
hd_ob1krf	dls	ws7-krf	58624
hd_ob1krf	rds	ws2-krf	60422
hd_ob1krf	oper	ds1-krf	182
sysmaster	ajm	ws1-krf	60611

display free dbspace query

```
-- dbsfree.sql - display free dbspace like Unix  "df -k " cmd

select  name[1,8] dbspace, sum(chksize) Pages_size,

sum(chksize) - sum(nfree) Pages_used,
sum(nfree) Pages_free,
round ((sum(nfree)) / (sum(chksize)) * 100, 2) percent_free
from    sysdbspaces d, syschunks c
where   d.dbsnum = c.dbsnum
group by 1
order by 1;
```

Example of dbsfree.sql

dbspace	pages_size	pagees_used	pages_free	percent_free
dbarch	1002000	766383	235617	23.51
dbgis	1002000	20495	981505	97.95
dbihfs	1274500	404313	870187	68.28
dbtmp	20000	661	19339	96.70
rootdbs	100000	61351	38649	38.65
textdbs	5000	1069	3931	20.85
textndx	25000	3249	21751	87.00

List all databases on server query

```
-- dblist.sql - List all databases, owner and logging status
select dbinfo("DBSPACE", partnum) dbspace,
name database, owner, is_logging, is_buff_log
from sysdatabases
order by dbspace, name;
```

Example of dblist.sql

dbspace	database	owner	is_logging	is_buff_log
dbarch	hdkrf_old	oper	1	1
dbihfs	dc5_22sss	oper	1	1
dbihfs	hd_ob1krf	oper	1	1
dbkrf2	vdb1_1krf	ajm	1	1
dbkrf3	areal_val	oper	0	0
dbsws	fastetc	oper	1	1
textdbs	fxatext	informix	1	1

tabextents query

```
-- tabextent.sql - List tables, number of extents and size of table
select  dbsname, tabname,
count(*) num_of_extents,
sum( pe_size ) total_size
from systabnames, sysptnext
where partnum = pe_partnum
group by 1, 2
order by 3 desc, 4 desc;
```

Example of tabextents.sql #1

dbsname	tabname	num_of_extents	total_size
areal_val	ro	111	30520
areal_val	qpf	111	30520
areal_val	map	111	30520
areal_val	mapx	110	30392

Example of tabextents.sql #2

dbsname	tabname	num_of_extents	total_size
hd_ob1krf	unkstnvalue	43	23048
hd_ob1krf	dpaadapt	18	3508
hd_ob1krf	height	14	67328
hd_ob1krf	precip	13	67328
hd_ob1krf	rwbiasdyn	13	1640
hd_ob1krf	cstprecip	12	57664
hd_ob1krf	temperature	9	39128
hd_ob1krf	curprecip	8	8320
hd_ob1krf	location	8	5640

table I/O activity query

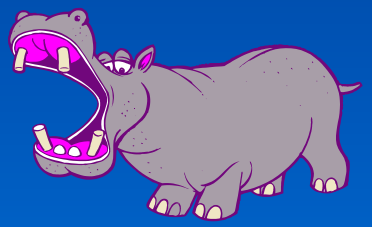
```
-- tabprof.sql
select dbsname, tabname, isreads, bufreads,
pagreads
-- uncomment the following to show writes
-- iswrites,
-- bufwrites,
-- pagwrites
-- uncomment the following to show locks
-- lockreqs,
-- lockwts,
-- deadlks
from sysptprof   order by isreads desc;
```

Example of tabprof.sql

dbsname	tabname	isreads	bufreads	pagreads
hd_ob1krf	coopcomms	0	0	0
hd_ob1krf	cooprecip	0	0	0
hd_ob1krf	coopspons	0	0	0
hd_ob1krf	damtypes	0	0	0
hd_ob1krf	dcpowner	0	0	0
hd_ob1krf	gagemaint	0	0	0
hd_ob1krf	gageowner	0	0	0
hd_ob1krf	gagetype	0	0	0
hd_ob1krf	hsa	0	0	0
hd_ob1krf	network	0	0	0
hd_ob1krf	proximity	0	0	0

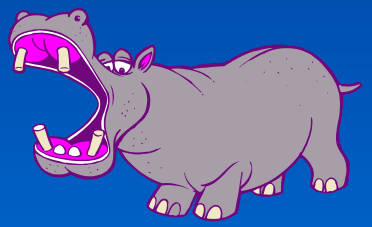
onmonitor

- Another way to see similar information as some of the queries just shown
- Do not have to be user Informix but must be on ds1 in order to use onmonitor



Other Commands & SQL

- finderr command
- dbload command
- oncheck and onstat commands
- update statistics
- set explain on sql
- onunload and onload commands



finderr Command

finderr searches the file of error message explanations distributed with Informix products and copies the text of one or more error messages to the standard output. If an unsigned number is given, a negative sign is assumed.

Examples:

```
finderr  327      (looks for message number -327)
finderr -327      (looks for message number -327)
finderr +1234     (looks for message number 1234)
finderr -233 107 113 134 143 144 +1541 | more
```

dbload Command

At the term window prompt:

dbload -c cmd_obs -d vdb1_1krf -l MIDNIGHT.err -n 5 -e 20000

Contents of the cmd_obs file:

FILE DATA1 DELIMITER '|' 9;
INSERT into obsht200301_1;
FILE DATA2 DELIMITER ‘|’ 9;
INSERT into obsht200212_1;

oncheck & onstat commands

oncheck -ci – check indexes for specified database/table

oncheck -pe – print detailed extents information

oncheck -pt – print information for specified table

onstat - – print version, status, uptime and memory usuage

onstat -u – print user threads

onstat -d – print dbspaces and chunks

onstat -g sql ‘session #’ – print sql information for session #

Update Statistics

- The update statistics statement is used to update system catalog with information that is used to determine optimal query plans.
- Update statistics can be performed on an entire database, individual table and procedures.
- Update statistics has 3 modes
  - low
  - medium
  - high

Update Statistics Low

Gathers very basic stats, which include:

- the number of rows in the table,
- the number of levels,
- number of leaf nodes, and number of unique keys in each index,
- second highest and second lowest value for each column,
- if DROP DISTRIBUTIONS is specified then any existing data distributions from previous MEDIUM or HIGH runs on the table(s) are deleted.

It is recommended that update statistics low be run as often as necessary to ensure that the statistic for the number of rows is as up-to-date as possible.

Update Statistics Medium or High

- Data distributions the accuracy of which depends on the level of stats and the confidence (for MEDIUM) and resolution (for either) you may specify.

- The difference is that HIGH counts every row while MEDIUM uses sampling the size of the sample depends on the confidence setting.

- If distributions only is not specified then the same stats as low are also gathered.

Update Statistics Strategy

1. Run UPDATE STATISTICS MEDIUM for all columns in a table that do not head an index. This step is a single UPDATE STATISTICS statement. The default parameters are sufficient unless the table is very large, in which case you should use a resolution of 1.0, 0.99. With the DISTRIBUTIONS ONLY option, you can execute UPDATE STATISTICS MEDIUM at the table level or for the entire system because the overhead of the extra columns is not large.

2. Run UPDATE STATISTICS HIGH for all columns that head an index. For the fastest execution time of the UPDATE STATISTICS statement, you must execute one UPDATE STATISTICS HIGH statement for each column. In addition, when you have indexes that begin with the same subset of columns, run UPDATE STATISTICS HIGH for the first column in each index that differs.

Update Statistics Strategy cont.

3. For each multicolumn index, execute UPDATE STATISTICS LOW for all of its columns. For the single-column indexes in the preceding step, UPDATE STATISTICS LOW is implicitly executed when you execute UPDATE STATISTICS HIGH.

4. For small tables, run UPDATE STATISTICS HIGH.

Because the statement constructs the index information statistics only once for each index, these steps ensure that UPDATE STATISTICS executes rapidly.

Update Statistics

For More Information

- Informix Guide to SQL: Syntax (Feb 1998)
- Performance Guide for IDS Version 7.3 (Feb 1998)

set explain on

Use this sql command with the query being tested to see how efficient it is.

Excellent Reference is Informix Tech Notes Vol 11, Issue 2, 2001,

Article “Optimizing Informix SQL: Examples and Analysis”

Sample SQL:

```
set explain on;
select lid,obstime,value from height
where pe=”HG” and date(obstime) = “01-28-2003"
order by 2;
```

set explain on sample output

```
select lid,obstime,value from height
where lid="KCDM7" and pe="HG" and date(obstime) = "01-28-2003"
order by 2
```

Estimated Cost: 422

Estimated # of Rows Returned: 117

Temporary Files Required For: Order By

1) oper.height: INDEX PATH
Filters: DATE (oper.height.obstime) = 01/28/2003
(1) Index Keys: lid pe dur ts extremum obstime
Lower Index Filter: (oper.height.lid = 'KCDM7' AND oper.height.pe = 'HG')

onunload/onload commands

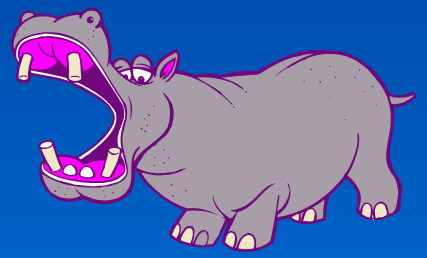
Although dbexport/dbimport is probably more familiar to NWS personnel, it has some disadvantages over onunload/onload. Dbexport requires an exclusive lock on the database, converts the data to ASCII text file format and can be very slow.

Onunload requires only a shared lock, similar to other backup commands and writes the data out in binary format and because of how it performs this write is much faster.

Onunload/onload should not be used for data migration to other platforms and versions of Informix.

3 Approaches to Managing Extents

- ALTER INDEX TO CLUSTER
- ALTER NEXT EXTENT/ALTER FRAGMENT
- Unload data. Redefine Table & Reload Data



Approach #1

alter index precip_time_ind to cluster;

alter index precip_time_ind to not cluster;

create cluster index bozo2 on latestobsvalue(obstime);

drop index bozo2;

Approach #2

```
alter table mapx modify next size 2400;

alter fragment on table mapx init in dbkrf3;
```

Two Methods for Calculating Extent Sizes

- Rule of Thumb for Next Extent Size
  multiply the number of extents by the next extent size currently used by that table and divide by 2. (courtesy of Patrick Sneeringer, WGRFC)

- Extent_calc.c application
  this apps was developed from information in Informix training manual “Managing and Optimizing IDS Databases”
  (courtesy of Paul Tilles, OHD/HL)

extents_calc.c

(1) Determine the normal number of rows in the table - checking the db_purge logs can give an estimate of how many records are placed in the table daily -multiply this estimate by the number of days of data to be held in the table

(2) Calculate the row size (bytes) and add 4 bytes for slot entry

(3) Determine how many rows will fit on a page and round down
(1 page = 2048 bytes, usable space on a page = 2020 bytes)
rows per page = 2020 / (size from step (2))

(4) Calculate how many pages are needed and round up
= (number of rows from step (1)) / (number of rows per page from step (3))

(5) Calculate total space needed
= (number of pages from step (4)) * (2Kb)

(6) Increase result by 30% to account for indexes & future growth of table

The next extent size can be estimated using these steps with an estimated number of additional rows in step (1). Rule of thumb is 10-25% of well sized initial size.

Approach #3

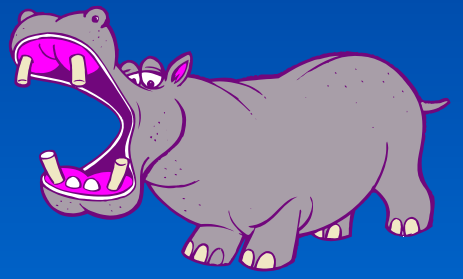
- unload the data from the table
- use the dbschema cmd to get table definition
- drop the table
- recreate the table with larger extents and without indexes
- reload the data
- recreate indexes

Resources

- Other NWS people
- Informix Manuals
- INFORMIX Handbook
- other Informix Books
- IIUG
- Informix news group: comp.databases.informix
- IBM Informix website

Some of My Favorite Websites

- www.iiug.org/home.html
- advancedatatools.com/articles.html
- www.markscranton.com/informix/informix.htm
- www.prstech.com/



The End